

**Logical-physical address remapping execution method in NAND-type flash memory, involves writing additional data to new cell block and deleting predetermined block**

**Patent Assignee: SAMSUNG ELECTRONICS CO LTD**

**Inventors: KIM B; LEE G; KIM B S; LEE G Y**

Patent Family							
Patent Number	Kind	Date	Application Number	Kind	Date	Week	Type
US 20020041517	A1	20020411	US 2001854920	A	20010515	200245	B
EP 1197868	A2	20020417	EP 2001303384	A	20010411	200245	
JP 2002123421	A	20020426	JP 2001123320	A	20010420	200245	
US 6381176	B1	20020430	US 2001854920	A	20010515	200245	
CN 1348191	A	20020508	CN 2001117722	A	20010429	200253	
KR 2002028624	A	20020417	KR 200059731	A	20001011	200268	

**Priority Applications (Number Kind Date):** KR 200059731 A ( 20001011)

Patent Details					
Patent	Kind	Language	Page	Main IPC	Filing Notes
US 20020041517	A1		15	G11C-016/04	
EP 1197868	A2	E		G06F-012/02	
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR					
JP 2002123421	A		12	G06F-012/02	
US 6381176	B1			G11C-016/04	
CN 1348191	A			G11C-016/10	
KR 2002028624	A			G11C-016/00	

**Abstract:**

US 20020041517 A1

NOVELTY A predetermined physical cell with free block is searched, if a specific block in the cell is in valid state. A cyclic order of the block status is established. The status of the free block is changed to status next to the current status of the specific block. New data and additional information is written to the other block and specific block is deleted.

DETAILED DESCRIPTION An INDEPENDENT CLAIM is included for flash memory.

USE For logical-physical address remapping in NAND-type flash memory (claimed).

ADVANTAGE Enables reusing the spaces in memory by deleting predetermined blocks and updating, and hence reduces write operation.

DESCRIPTION OF DRAWING(S) The figure shows the relationship between blocks explaining write and rewrite operations in flash memory.

Derwent World Patents Index

© 2004 Derwent Information Ltd. All rights reserved.

Dialog® File Number 351 Accession Number 14604671

# (19) 대한민국특허청(KR) (12) 공개특허공보(A)

(51) Int. Cl.<sup>7</sup>  
G11C 16/00

(11) 공개번호 특2002-0028624

(43) 공개일자 2002년04월 17일

(21) 출원번호	10-2000-0059731
(22) 출원일자	2000년 10월 11일
(71) 출원인	삼성전자 주식회사
(72) 발명자	경기 수원시 팔달구 매탄3동 416 김범수 경기도 안양시 동안구 평촌동 초원마을 대림아파트 203동 1105호 이귀영 서울특별시 노원구 상계3동 107번지 불암동 아아파트 105동 1005호
(74) 대리인	이영필, 최흥수, 이해영

심사청구 : 없음

(54) 플래시메모리를 위한 재사상 제어방법 및 그에 따른 플래시 메모리의 구조

## 요약

본 발명은 부분 기록을 최소화하면서 블록과 유니트의 상태를 기록하고 처리할 수 있는 플래시메모리의 재사상 제어 방법 및 그에 따른 플래시메모리의 구조를 개시한다. 본 발명에 따른 방법은, (a)소정 블록에 대한 사상 정보를 토대로 소정의 물리적인 유니트를 찾는 단계; (b) (a)단계에서 찾아진 물리적인 유니트에 소정 블록이 유효한 상태로 존재하면, 블록의 상태가 미정인 다른 블록을 찾는 단계; (c) 다른 블록의 상태를 소정 블록에 설정되어 있는 상태의 다음 상태로 변경하는 단계; (d) 다른 블록에 새로운 데이터와 논리적 블록 번호와 같은 부가 정보를 기록하는 단계; (e) 소정 블록의 상태를 삭제 상태로 변경하는 단계를 포함한다. 따라서, 부분기록 회수에 제한이 있는 플래시메모리에 대해 블록이나 유니트의 기록상태를 관리할 수 있다.

## 대표도

## 도4

## 명세서

### 도면의 간단한 설명

도 1은 플래시메모리의 블록 및 유니트 구성 예 및 그에 따른 테이블 구성도이다.

도 2는 도 1에 도시된 바와 같이 데이터를 기록한 플래시메모리에 대한 블록단위 삭제 시 플래시메모리의 블록 및 유니트 구성 예 및 그에 따른 테이블 구성도이다.

도 3은 도 2에 도시된 바와 같이 데이터를 기록한 플래시메모리에 대한 유니트 단위 삭제 시 플래시메모리의 블록 및 유니트 구성 예 및 그에 따른 테이블 구성도이다.

도 4는 본 발명에 따른 플래시메모리를 위한 재사상 제어방법에 있어서 기록 및 수정 동작을 설명하기 위한 블록간 관계 예시도이다.

도 5는 NAND타입 플래시메모리의 물리적인 유니트(PU)의 포맷 예시도이다.

도 6은 본 발명에 따른 플래시메모리를 위한 재사상 제어방법에 있어서 블록 기록과정에 대한 동작 흐름도이다.

도 7은 본 발명에 따른 플래시메모리의 삭제유니트의 포맷도이다.

도 8은 본 발명에 따른 플래시메모리를 위한 재사상 제어방법에 있어서 재생연산과정을 설명하기 위한 유니트간 관계 예시도이다.

도 9는 본 발명에 따른 플래시메모리를 위한 재사상 제어방법에 있어서 재생연산과정에 대한 동작 흐름도이다.

### 발명의 상세한 설명

#### 발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 블록단위로 플래시메모리에 데이터를 기록하거나 읽는 방법 및 그에 따른 플래시메모리의 구조에 관한 것으로, 특히, 부분적으로 기록 회수가 제한되어 있는 플래시메모리에/로부터 데이터를 기록하거나 읽을 때, 논리적인 주소와 물리적인 주소간의 재사상(remapping)을 제어하는 방법과 그에 적합한 플래시메모리의 구조에 관한 것이다.

플래시메모리는 기존의 RAM(Random Access Memory)이나 비휘발성 저장장치, 마그네틱 디스크 등과 마찬가지로 특정 위치에 저장된 데이터를 임의로 접근할 수 있다.

그러나, 데이터를 수정하거나 삭제하는 방법이 상술한 다른 저장장치들과 다르다. 즉, 블록단위 접근을 위하여 일정한 크기의 블록으로 분할된 플래시메모리의 소정 블록에 데이터를 한 번 기록한 뒤, 이를 수정하거나 삭제하고자 할 때에는 해당 블록을 포함하고 있는 유니트를 수정하거나 삭제하여 한다. 플래시메모리에서 블록은 물리적으로 연속된 주소를 갖는 바이트로 구성된 것으로 플래시메모리에 대한 연산의 기본 단위이다. 유니트는 복수개의 블록들로 구성된 것으로, 물리적으로 한 번에 삭제 또는 수정할 수 있는 기본 단위이다.

이러한 플래시메모리의 운영 특성 때문에 삭제 회수를 감소시켜서 데이터 기록 및 갱신 효율을 높이고, 고장이 발생해도 데이터가 손실되지 않게 하기 위해 일반적으로 블록(또는 섹터) 재사상(remapping) 기법을 사용하고 있다.

재사상 기법은, 수정 또는 삭제로 인하여 플래시메모리에 기록된 데이터의 물리적인 블록번호(Physical Block Number, PBN이라고 약하기도 함)가 변경되어도 동일한 논리적인 블록번호(Logical Block Number, LBN이라고 약하기도 함)로 플래시메모리에 기록된 데이터에 대한 접근이 가능하도록, 해당 데이터에 대한 논리적인 블록번호와 물리적인 블록 번호(PBN)간의 사상 정보를 관리하는 것이다.

기존의 재사상 기법으로는 수정하고자 하는 데이터가 발생되면, 플래시메모리의 해당 데이터의 물리적인 블록번호를 포함하고 있는 유니트에서 비어 있는 물리적인 블록을 찾아서 기록하고, 해당되는 데이터의 논리적인 블록번호와 물리적인 블록번호간의 사상 정보를 갱신한 뒤, 이전의 물리적인 블록의 상태 정보에 삭제 표시를 한다. 따라서 사용자는 동일한 논리적인 블록 번호를 사용하여 데이터에 접근할 수 있게 된다.

예를 들어 플래시메모리에 도 1에 도시된 바와 같이 데이터가 기록된 상태에서 사용자가 논리적인 블록 번호가 '3'인 블록에 기록된 데이터를 수정하고자 할 때, 도 2에 도시된 바와 같이 해당되는 유니트의 상태가 변경된다. 즉, LBN-to-LUN 테이블을 참조하면, 논리적인 블록 번호 '3'에 대응되는 논리적인 유니트 번호는 '2'이다. 이 논리적인 유니트 번호 '2'를 이용하여 LUN-to-PUN 테이블을 참조하면, 대응되는 물리적인 유니트 번호는 '1'임을 알 수 있다. 따라서 물리적인 유니트 번호가 '1'인 유니트에서 빈 블록을 찾는다. 도 1에 도시된 바와 같이 '4'번 블록이 비어 있는 블록이므로, 논리적인 블록 번호 '3'에 해당되는 수정하고자 하는 데이터는 도 2에 도시된 바와 같이 빈 블록에 기록하고, 블록할당 맵(BAM)의 사상 정보를 갱신한 뒤, 블록할당 맵(BAM)에서 이전의 물리적인 블록이었던 '1'에 대한 상태 정보에 삭제 표시를 한다.

그러나, 해당되는 유니트에 사용되지 않는 물리적인 블록이 많아지면, 플래시메모리의 사용 효율이 낮아지므로 기존에는 도 3에 도시된 바와 같이 사용되는 부분만을 다른 유니트로 이동(transfer)시키고, 해당되는 물리적인 유니트 번호를 상기 다른 유니트의 물리적인 유니트 번호로 변환함으로써 데이터를 이동하더라도 동일한 논리적인 유니트 번호로 접근할 수 있도록 하였다.

이와 같은 재사상 기법은 유니트내에 빈 블록이 존재하거나 플래시메모리 상에 빈 유니트가 존재하는 한 실질적인 삭제 연산을 수행하지 않는다. 그러나, 삭제된 블록이 증가하면 플래시메모리에서 사용할 수 없는 영역이 증가하는 것을 의미하므로, 삭제된 블록이 있는 부분을 재사용하기 위한 방법이 마련되어야 할 것이다.

또한, 쓰기 연산을 수행하거나 유니트의 재사용 연산을 수행하는 도중에 전원 중단 등의 장애가 발생하면, 사용자의 데이터나 재사상을 위해 사용되는 데이터가 손실될 수 있다. 따라서 기존에는 이를 대비하기 위하여 블록과 유니트에 상태 정보를 기록하고, 연산 수행시 상태정보를 적절히 수정함으로써, 복구 연산을 수행하도록 하고 있다.

상술한 바와 같이 기존의 재사상 기법은 데이터뿐만 아니라 사상 정보를 저장하고 관리하는 데도 저장공간과 시간이 소요되므로, NAND타입의 플래시메모리와 같이 부분적인 쓰기 연산의 회수(Number of program cycles in the same page, 이하 Nop라고 약함)가 제한되는 플래시메모리에는 적용하기 어렵다. 이는 상술한 NAND타입의 플래시메모리와 같은 플래시메모리는 장애로 인한 복구를 위하여 사용되는 블록과 유니트의 상태 정보를 여러 번 변경시킬 수 있는 여지가 별로 없기 때문이다.

예를 들어, 소정 블록의 상태가 미정(FF)→할당(8F)→기록중(4F)→기록(2F)→삭제중(0F)→삭제(00)와 같은 절차로 블록의 상태를 표시하거나 미정(FF)→복사중(8F)→유효(4F)→이동중(2F)과 같은 절차로 유니트의 상태를 표시하려면 물리적으로 동일한 위치에 대해 최소한 5회 내지 3회의 쓰기 연산을 수행해야 한다. 뿐만 아니라 유니트나 블록에 사용자 데이터나 에러정정코드(ECC) 등의 경로를 기록하는 것까지 생각한다면, 상술한 바와 같이 Nop가 5회 또는 3회 등으로 제한된 플래시메모리에 기존의 재사상 기법을 적용하기는 어렵다.

#### 발명이 이루고자하는 기술적 과제

본 발명은 상술한 문제를 해결하기 위하여, 부분 기록을 최소화하면서 블록과 유니트의 상태를 기록하고 처리할 수 있는 플래시메모리의 재사상 제어 방법 및 그에 따른 플래시메모리의 구조를 제공하는데 그 목적이 있다.

본 발명의 다른 목적은 순환계수(wrap-count)를 사용하여 최소한의 부분기록으로 상태정보를 표시하는 플래시메모리의 재사상 제어 방법 및 그에 따른 플래시메모리의 구조를 제공하는데 그 목적이 있다.

## 발명의 구성 및 작용

본 발명이 이루고자 하는 목적들을 달성하기 위하여 본 발명에 따른 방법은, 적어도 1개의 블록을 구비하고 있는 유니트 단위로 영역이 구분된 플래시메모리의 재사상 제어 방법에 있어서, (a)소정 블록에 대한 사상 정보를 토대로 소정의 물리적인 유니트를 찾는 단계;(b) (a)단계에서 찾아진 물리적인 유니트에 소정 블록이 유효한 상태로 존재하면, 블록의 상태가 미정인 다른 블록을 찾는 단계;(c) 다른 블록의 상태를 소정 블록에 설정되어 있는 상태의 다음 상태로 변경하는 단계;(d)다른 블록에 새로운 데이터와 논리적 블록 번호와 같은 부가 정보를 기록하는 단계;(e) 소정 블록의 상태를 삭제 상태로 변경하는 단계를 포함하는 것이 바람직하다.

특히, 상기 방법은, (f) (a)단계에서 찾아진 물리적인 유니트에 소정 블록이 존재하지 않거나 소정 블록이 삭제 상태로 존재하면, 소정 블록의 상태를 미정상태로 설정한 뒤, 블록의 상태가 미정인 다른 블록을 찾는 단계와 (g) (f)단계 수행결과, 다른 블록이 존재하지 않으면, 물리적인 유니트에 대해 재생연산을 수행하는 단계를 더 포함하는 것이 바람직하다.

상기 (g)단계에서 이루어지는 재생연산은, (g1)재생할 소스 유니트와 이동 유니트를 결정하는 단계;(g2) 소스 유니트의 상태를 이동중으로 설정하고, 이동 유니트의 상태를 복사중으로 설정하는 단계;(g3) 소스 유니트내의 유효한 블록들, 관련 메타 및 헤더 정보를 복사하는 단계;(g4) 복사중으로 설정되어 있는 이동 유니트를 유효 상태로 설정하면서 삭제 회수(마모도 값)를 1회 증가시키는 단계;(g5) 소스 유니트를 삭제하면서 (g4)에서 결정된 삭제 회수를 소스 유니트에 기록하는 단계를 포함하는 것이 바람직하다.

상기 목적들을 달성하기 위한 플래시메모리의 구조는, 유니트의 첫 번째 블록은 논리적인 유니트 번호와 유효한 상태여부, 이전 블록의 삭제 회수를 포함하는 정보의 기록을 위해 할당하고, 유니트의 두 번째 블록은 복사중/이동중 여부와 삭제 회수(마모도)를 포함하는 정보의 기록을 위해 할당하도록 구성되고, 유니트의 데이터가 기록되는 블록의 스페어 어레이의 소정 영역에는 순환계수방식으로 결정된 블록상태를 나타내는 정보가 기록되도록 구성되는 것이 바람직하다.

이하, 첨부된 도면을 참조하여 본 발명을 상세히 설명한다.

먼저, 플래시메모리 재사상 제어에 있어서 삭제 단위로 사용하는 것은 삭제 유니트이다. 삭제 유니트는 플래시메모리의 물리적 삭제 블록(도 1 내지 도 3에 도시되어 있는 PUN1, PUN2와 같은 PUN)과 일치할 수도 있고, 여러 개의 삭제 블록으로 구성될 수도 있다.

각 삭제 유니트에는 물리적 유니트 번호(PUN)와 논리적인 유니트 번호(LUN)가 부여된다. 물리적 유니트 번호는 삭제 유니트의 물리적 순서에 따라 할당되며, 논리적 유니트 번호는 유니트의 논리적 순서를 나타내는 번호이다. 플래시메모리를 처음 사용할 때, 도 1 내지 도 3에 도시된 바와 같은 물리적 유니트 번호(PUN)와 논리적 유니트 번호(LUN)간의 사상을 나타내는 테이블(LUN-to-PUN)이 플래시메모리 상 또는 플래시메모리와 별도로 구비되어 있는 메모리에 구성된다. 이 테이블은 데이터의 삭제나 변경, 재생 등의 연산으로 인해 사상 관계가 변할 때마다 변경된다.

삭제 유니트는 도 1 내지 도 3에 도시된 PUN1, PUN2에 도시된 바와 같이 다시 작은 크기의 블록(또는 섹터)으로 분할된다. 이 블록은 플래시메모리에 대한 연산의 기본단위로 사용된다. 플래시메모리에서 모든 블록의 크기는 동일하다.

삭제 유니트의 삭제 유니트 헤더(Erase Unit Header, EUH)에는 플래시메모리 전체에 대한 정보(예를 들어 블록의 크기, 불량 블록 맵(bad block map) 등)과 그 유니트의 관리에 필요한 정보(예를 들어 논리적 유니트 번호, 마모도 등)가 기록된다. 상기 삭제 유니트 헤더는 하나 또는 그 이상의 블록을 차지할 수 있다.

삭제 유니트 헤더 다음에는 도 1 내지 도 3의 PUN1, PUN2에 도시된 바와 같이 사용자 데이터와 블록 할당 맵(Block Allocation Map, BAM)이 기록된다. 블록 할당 맵은 하나 또는 그 이상의 블록에 모아서 기록될 수도 있고, NAND 타입 플래시메모리의 스페어 어레이(spare array)와 같이 특수한 공간에 블록별로 나누어 기록될 수도 있다.

블록할당 맵(BAM)에는 그 유니트에 속한 블록들에 대한 정보(논리적 블록번호, 상태 등)가 기록된다. 논리적 블록번호는 사용자가 생각하는 블록의 주소이다. 따라서 블록할당 맵은 논리적 블록 번호와 그 블록이 실제로 기록된 물리적 위치간의 사상을 나타낸다. 블록에 기록되는 데이터가 변경될 때는 블록할당 맵의 내용, 즉, 논리적 블록번호와 물리적 블록번호 간의 사상 정보와 상태 정보도 변경된다.

변경된 블록이 많아지면, 플래시메모리에 못쓰게 된 부분이 많아지므로 재생연산을 적절히 수행하여 삭제된 블록이 차지하던 공간을 다시 쓸 수 있게 해야 한다. 기존의 재사상 제어방식에서도 이에 대해 설명하고 있기는 하지만 NAND 타입 플래시메모리와 같이 부분 기록의 회수가 제한된 플래시메모리에 대해 상술한 재생 연산을 어떻게 수행하는지에 대해서는 명시하고 있지 않다.

본 발명에서는 블록의 상태 정보를 순환계수(wrap-count)를 사용하여 표시하는 방법을 제시한다. 이 방법을 사용하면 최소한의 부분 기록 회수로 블록의 상태를 변화시키고, 고장이 발생할 경우에 해당 블록의 데이터를 회복할 수 있다. 또한, 순환계수의 변화순서를 이용하여 이전 데이터와 새 데이터를 구분할 수 있어서 롤백(roll-back)과 롤포워드(roll-forward) 등 여러 가지 회복방법을 사용할 수 있다. 롤백 회복 방법은 새 데이터를 이용하여 데이터를 회복하는 것이고, 롤 포워드는 이전 데이터를 이용하여 데이터를 회복하는 것이다.

이와 같이 플래시메모리의 재사상 제어동작은 블록 기록, 검색, 고장에서의 회복 등으로 구분된다.

블록 할당 맵에서 각 블록의 상태는 크게 미정(free), 유효(valid), 삭제(deleted)로 나뉜다. 이 중에서 '유효'는 다시 세 가지 상태로 구분된다. 예를 들어 s0, s1, s2 상태로 구분된다. 유효한 상태의 세 값 사이에는 순환적인 순서 관계가 있다. 예를 들어 s0

(s1, s1

(s2, s2

(s0인 관계를 가질 수 있다. 유효한 상태는 이 순서에 따라 변화하므로 이것을 순환계수라고 한다. 즉, 블록의 처음 상태는 미정이다가 데이터가 처음 기록될 때 s0가 되며, 그 다음에 다시 기록될 때는 s1, s2, 다시 s0으로 차례로 변하게 된다.

이와 같은 상태정보를 이용하여 소정 블록에 데이터를 기록하는 과정은 도 4와 같다. 도 4는 미정 상태를 '1111'로 설정하고, 유효한 상태의 세 값 중 s0을 '1110'으로 설정하고, s1을 '1100'으로 설정하고, s2를 '1000'으로 설정하고, 삭제를 '0000'으로 설정한 예이다. 따라서 블록의 상태는 상기 5가지 경우의 값이 순환되면서 설정된다. 그리고 도 4는 도 5에 도시된 바와 같이 물리적인 유니트가 구성된 NAND타입 플래시메모리에 적용한 예이다.

도 5에 도시된 바와 같이 NAND타입 플래시메모리의 물리적인 유니트(PU)내의 블록은 메인 어레이(main array)와 스페어 어레이(spare array)로 구성된다. 메인 어레이에는 실질적인 데이터(data)만 기록된다. 스페어 어레이에는 논리적인 섹터 번호(logical sector number, lsn), 순환계수(wrap count, cnt), 논리적인 섹터 번호(lsn)과 순환계수(cnt)의 에러정정 코드(ecc\_lsn)와 데이터에 대한 에러 정정 코드(ecc\_data) 등이 기록된다. 이 때, 블록을 페이지(page)라고도 표현한다.

NAND타입 플래시메모리 데이터 시트에서 흔히 언급하는 '블록'은 한번에 삭제할 수 있는 물리적인 단위로, 본 실시 예에서 언급되고 있는 '유니트'에 해당한다고 볼 수 있다. 그러나 실제로 구현시에는 상기 유니트를 여러 블록으로 구성할 수도 있다.

먼저, 사상 정보를 이용하여 해당 블록이 속한 논리적 유니트 번호와 물리적 유니트 번호를 결정한다. 결정된 물리적 유니트의 블록 할당 맵에서 해당 블록의 상태를 체크한 결과 미정(1111)이면, 해당 블록의 상태를 s0(=1110)으로 표시한 뒤, 데이터(data)와 논리적 블록번호(lsn) 등의 정보를 기록한다. 미정인 상태의 블록은 논리적 블록번호(lsn)도 유효하지 않은 어떤 값을 갖는다.

해당 블록의 상태가 유효인 경우에는 블록의 상태가 미정(1111)인 블록을 일반적인 사상 방식에 따라 찾는다. 그리고, 검색된 블록의 상태를 이전의 블록 상태의 다음 상태로 변경한다. 즉, 상술한 기존 블록의 상태가 s0(1110)이므로 s1(=1100)로 변경한다. 블록 검색은 기존의 재사상 제어방식에서 이루어진 것과 동일하게 이루어진다.

그리고 나서 새 블록(블록의 상태가 s1(1100)으로 설정된 블록)에 새로운 데이터(data)와 논리적 블록번호(lsn) 등을 기록한다. 그리고, 이전 블록의 상태를 삭제(0000)로 변경한다.

그 다음, 해당 블록의 상태가 s1(=1100)이고, 해당 유니트에 미정상태(1111)로 설정되어 있는 블록이 존재하는 경우에, 미정상태로 설정되어 있는 블록의 상태를 s2(=1000)로 설정하고, 블록의 상태가 s2로 설정된 블록에 데이터(data)와 논리적인 블록 번호(lsn) 등의 정보를 기록하고, 이전 블록의 상태는 삭제(=0000)로 변경한다.

또한, 해당 블록의 상태가 s2(=1000)이고, 해당 유니트에 미정상태(1111)로 설정되어 있는 블록이 존재하는 경우에, 미정상태로 설정되어 있는 블록의 상태를 s0(=1110)으로 설정하고, 블록의 상태가 s0으로 설정된 블록에 데이터(data)와 논리적인 블록 번호(lsn) 등의 정보를 기록하고, 이전 블록의 상태는 삭제(=0000)로 변경한다.

상술한 블록의 상태 변화는 도 4에 도시된 화살표로 표시된 순서로 따라갈 경우에, 충분히 이해할 수 있을 것이다. 이와 같이 블록에 대한 기록 및 수정 처리를 할 경우에, 1 블록에 대해 3회의 부분 기록으로 데이터를 기록할 수 있다. 삭제된 상태인 경우에, 데이터와 논리적 블록 번호 등 상태 정보를 제외한 다른 데이터는 모두 유효한 것으로 가정한다면 이는 실제로 2회의 부분 기록으로 기록 연산을 수행하는 것과 같다.

그리고, 상술한 작업 도중에 401 내지 407 단계에서 전원이 오프 되거나 기타 다른 이유로 장애가 발생한 경우에 데이터를 기록하고자 했던 대상 블록의 상태는 삭제(=0000)로 설정하는 회복작업을 수행한다. 그 이외의 작업 도중에는 상술한 바와 같이 장애가 발생된 경우에는 별다른 회복작업을 수행할 필요가 없다.

다시 말해서 블록에 데이터를 기록하던 중에 장애가 발생한 경우, 미정이거나 삭제 상태인 블록에 대해서는 어떠한 회복작업도 해 줄 것이 없다. 하나의 논리적 블록 번호에 대해 유효한 상태인 블록이 하나 밖에 없을 때도 역시 어떠한 회복작업도 해 줄 것이 없다.

문제는 동일한 논리적 블록 번호를 가진 유효한 블록이 두 개 존재하는 경우이다. 기존에는 이 같은 경우에 이전 블록과 새 블록을 구별할 수 없었으나 본 발명과 같이 순환계수를 사용하게 되면 이전 블록과 새 블록을 구별할 수 있다. 회복 시에 이전 블록을 삭제할 것인지 새 블록을 삭제할 것인지에 관한 정책은 응용의 필요에 따라 결정할 수 있다.

즉, 블록 데이터를 기록하던 중에 장애가 발생되면, 현재 하나의 논리적 블록 번호에 대해 유효한 상태를 갖는 블록이 두 개 존재하는 지를 체크한다. 체크결과, 유효한 상태를 갖는 블록이 하나만 존재하는 경우에는 상술한 바와 같이 아무런 회복작업도 해 줄 것이 없다. 따라서 블록 데이터를 기록하던 작업을 중단하면 된다. 그러나, 유효한 상태를 갖는 블록이 두 개 존재하는 경우에, 블록에 할당되어 있는 순환계수의 값을 토대로 이전 블록과 새 블록을 구분하고, 이전 블록과 새 블록중 하나의 블록을 삭제한다. 도 4는 새 블록을 삭제하도록 구현한 예이다. 그러나, 이전 블록을 삭제하도록 구현할 수도 있다.

도 6은 상술한 재사상 제어 방법에 따라 블록을 기록 및 수정하고자 할 때의 동작 흐름도이다.

즉, 단계 601에서 해당 유니트에 해당 블록이 존재하는지를 체크한다. 체크한 결과, 존재하면 해당되는 블록의 상태가 유효상태인지를 체크한다. 체크결과 유효한 상태이면, 단계 603에서 해당 블록의 상태를

체크한다. 그 다음 단계 605에서 해당 유니트에 빈 블록이 존재하는 지를 체크한다. 체크방식은 블록의 상태가 미정으로 설정되어 있는 블록이 있는지를 검색하는 방식으로 이루어진다. 체크결과, 해당 유니트에 빈 블록이 존재하면, 단계 607에서 새로운 블록 상태 값을 결정한다.

새로운 블록 상태 값은 단계 603에서 체크한 해당 블록의 상태값을 고려하여 결정한다. 즉, 상술한 도 4에 대한 설명에서 언급한 바와 같이 이전의 블록의 상태가 s0에 해당되면, 새로운 블록 상태 값은 s1로 설정되고, 이전 블록의 상태가 s1에 해당되면, 새로운 블록 상태 값은 s2로 설정하고, 이전 블록의 상태가 s2에 해당되면, 새로운 블록 상태 값은 s0로 설정한다. 그리고, 이전 블록의 상태가 미정(free)상태이면, s0로 설정한다.

그 다음 단계609에서 단계 605에서 검색된 빈 블록에 단계 607에서 결정된 새로운 블록 상태값을 기록하고, 단계 611에서 새로운 블록 상태값을 기록한 새로운 블록에 해당되는 데이터를 기록하고, 단계 613에서 이전 블록의 상태를 삭제로 변경하고, 작업을 종료한다. 이 때, 이전 블록은 단계 603에서 상태가 체크된 블록이다.

한편, 단계 601에서 체크한 결과, 해당 블록이 존재하지 않거나 존재하기는 하나 유효한 블록이 아닌 경우(즉, 해당 블록의 상태가 삭제상태인 경우에), 단계 615에서 해당 블록을 미정상태로 설정한다. 이 때, 해당 블록이 존재하지 않는 경우에는 해당 블록에 대한 영역을 할당하면서 해당 블록의 상태를 미정상태로 설정한다. 그리고, 단계 605에서 체크한 결과, 빈 블록이 존재하면, 단계 607로 진행되어 상술한 바와 같이 새로운 블록 상태값을 결정한다.

그러나, 단계 605에서 체크한 결과, 해당 유니트에 빈 블록이 존재하지 않으면, 단계 617에서 재생연산을 수행한다. 재생연산은 후술할 도 7 내지 도 9에서 상세하게 설명하기로 한다.

재생연산 수행 후, 단계 619에서 해당 유니트에 빈 블록이 존재하는지를 체크한다. 체크결과, 해당 유니트에 빈 블록이 존재하면, 단계 607로 진행되어 새로운 블록 상태값을 결정한다. 그러나, 빈 블록이 존재하지 않으면, 단계 621에서 오류처리를 하고, 작업을 종료한다.

블록에서 데이터를 읽기 위해서는 먼저 사상 정보를 이용하여 해당 논리적 유니트 번호와 물리적 유니트 번호를 알아내고 해당 유니트의 블록 할당 맵을 통해 블록의 물리적 위치를 결정하여 데이터를 읽는다. 블록의 상태가 유효한 경우에만 데이터를 읽게 되어 있으며 유효한 상태의 블록이 존재하지 않으면 그 블록은 한 번도 기록되지 않았거나 삭제되어 사용되지 않고 있는 블록이므로 초기값(예를 들어 모두 0xFF)을 반환하게 된다. 검색은 기존의 재사상 제어방식에서 이루어진 것과 동일하다.

한편, 삭제된 블록들이 많이 들어 있는 유니트에 대해서는 재생 연산이 수행되는데 이 때도 유니트의 상태를 변경시킬 필요가 있다.

본 발명에서 예를 든 삭제 블록의 포맷은 도 7에 도시된 바와 같다.

즉, 도 7을 참조하면, 삭제 블록은 이전 유니트의 물리적인 유니트 번호(physical unit number of previous unit, xpun), 재생된(이전) 유니트의 마모도 레벨(wear level of reclaimed(previous) unit), 논리적인 유니트 번호(logical unit number, lun), 유효 플래그(valid flag, v), 불량 블록 테이블(bad block table, bb[b]), 마모도 레벨(wear level, cnt), 복사(copying(cp))/전송(transferring(xf)), 데이터(data) 등이 기록된다.

유니트의 상태 변화도 부분 기록 회수에 따라 제한된다. 블록에 적용하였던 순환계수기법을 유니트에도 적용할 수 있지만 삭제 유니트 헤더를 여러 블록에 기록하면서 상태 정보를 각 블록에 적절히 나누어 배치하는 방법을 사용할 수도 있다.

예를 들어, 유니트의 상태가 미정(free), 유효(valid), 복사중(being copied), 이동중(being transferred)으로 나뉜다면, 삭제 유니트 헤더를 도 8에 도시된 바와 같이 두 블록에 걸쳐서 기록하되 첫 번째 블록에는 미정과 유효인 상태를 두 번째 블록에는 복사중과 이동중인 상태를 기록할 수 있다. 이 때 미정인 상태라 하더라도 마모도 평준화(wear-leveling)를 위한 카운터가 기록되어야 한다는 점에 주의해야 한다.

자유 삭제 유니트의 상태는 미정이고, 그 외의 정상적인 유니트의 상태는 유효이다. 복사중과 이동중이라는 상태가 표시되고, 자유 삭제 유니트였다가 재생되는 블록으로부터 데이터를 이어 받는 유니트에는 복사중이라는 상태가 표시된다. 재생을 수행하기 전에 먼저 이전 유니트의 상태를 이동중(xf)이라고 표시하고 자유삭제 유니트의 상태를 복사중(cp)이라고 표시한다.

복사중인 유니트에 이동중인 유니트로부터 유효한 블록들을 복사한 뒤, 이동중인 유니트의 마모도 값과 논리적 유니트 번호 등을 복사한 뒤, 복사중인 유니트의 상태를 유효(v)로 표시한다. 그리고 나서 이동중인 유니트를 삭제하고, 새 유니트에 복사되었던 마모도 값(cnt)을 1만큼 증가시켜 유니트 헤더에 기록한다.

연산을 수행하던 중에 장애가 발생하면, 도 8에서 재시작(redo)으로 지정한 부분에서는 재 시작한 후에 적절한 회복 절차를 수행한다. 즉, 재생연산 도중의 일시적인 상태에 해당되는 '이동중', '복사중'인 경우에 장애가 발생하면, 재시작을 한다. 그러나, 도 8에서 재시작을 지정한 부분을 제외한 부분에서는 어떠한 회복 절차도 해 줄 것이 없다.

재생연산 수행중 발생한 장애에 대한 회복방법도 블록 기록 및 수정시 회복 방법과 비슷하다.

즉, 수행중이던 연산의 단계에 따라 적절히 재생 연산을 다시 수행한다. 예를 들어 논리적 유니트 번호가 같은 유니트가 두 개 존재하는데, 하나는 이동중이고, 다른 하나는 복사중이라면 복사중인 유니트를 삭제하고 마모도 값을 기록한다. 이동중인 유니트에 이전 유니트(복사중인 유니트)의 마모도가 아직 기록되지 않은 상태였다면 먼저 이전 유니트(복사중 유니트)에 기록되어 있는 이동중인 유니트에 대한 마모도 값을 이동중인 유니트에 복사한 뒤, 다음 연산을 수행하면 된다.

삭제 유닛의 상태는 종래 기술과 같이 고정된 위치에서 값을 여러 가지로 변경시켜 나타낼 수도 있고, 유닛 헤더를 여러 블록에 나누어 기록하며 각 상태에 대해 다른 위치를 지정하고 값을 0과 1 등으로 구분하여 상태값 할당여부를 결정할 수도 있다.

예를 들어 첫 번째 블록에 논리적 유닛 번호와 유효한 상태 여부, 이전 블록의 마모도 등을 기록하고, 두 번째 블록에 복사중/이동중 여부와 마모도 등을 기록하고, 도 8에서 제시한 것과 같은 순서에 따라 재생 연산을 수행한다면 3회의 부분 기록이면 충분하다. 상태 정보를 더 많은 블록에 분산시키거나 배치를 조절하면 부분 기록 회수를 더 줄일 수도 있다.

도 9는 본 발명에 따른 재사상 제어방법으로 블록을 재생하는 과정에 대한 동작 흐름도이다.

단계 901에서 재생할 삭제 유닛(소스 유닛)과 이동 유닛(자유 삭제 유닛)을 결정한다. 단계 902에서 이동 유닛의 삭제 회수가 임계치를 넘었는지를 체크한다. 임계치는 상술한 바를 토대로 할 때, 3으로 설정될 수 있다.

체크결과, 이동 유닛의 삭제 회수가 임계치를 넘지 않으면, 단계 903에서 소스 유닛을 '이동중' 상태로 변경한다. 단계 904에서 이동 유닛을 '복사중' 상태로 변경한다.

다음, 단계 905에서 소스 유닛에서 유효 블록과 관련 메타 데이터(논리적 블록 번호, ECC, 불량 블록 맵 등)를 이동 유닛으로 복사한다. 그리고, 단계 906에서 이동 유닛의 나머지 헤더 정보를 기록하고 유효 상태로 변경한다. 나머지 헤더 정보는 소스 유닛의 물리적인 유닛 번호와 삭제회수, 논리적 유닛 번호 등이다.

단계 907에서 소스 유닛을 삭제한다. 그리고, 단계 908에서 소스 유닛에 새로운 삭제 회수를 기록한다. 새로운 삭제 회수는 이전 삭제 회수에 1증가한 값(이전 삭제회수 +1)이다. 단계 909에서 사상 정보 등 메모리 내 자료 구조를 변경한다. 이 때, 변경은 갱신에 따른 것이다. 그리고, 단계 910에서 저장된 소스 유닛 번호가 있는지 체크한다. 체크결과, 있으면, 단계 901로 진행된다. 그러나, 체크결과, 없으면, 작업을 종료한다.

한편, 단계 902에서 체크한 결과, 이동 유닛의 삭제 회수가 임계치를 넘은 경우에는 단계 911에서 현재 소스 유닛의 번호를 저장한다. 이는 해당되는 이동 유닛에 소스 유닛의 데이터를 옮기는 것이 마땅치 않기 때문에, 후술할 단계 912에서 선택된 소스 유닛(삭제 유닛)을 이용한 재생 연산 처리 후, 바뀌는 이동 유닛을 이용하여 이전에 재생되지 못했던 소스 유닛을 재생할 때, 사용하기 위한 것이다.

그리고, 단계 912에서 삭제 회수가 최소인 삭제 유닛을 선택하고, 단계 903으로 진행되어 상술한 바와 같은 재생연산처리가 수행된다.

### 발명의 효과

상술한 바와 같이 본 발명에 의하면, 순환계수를 사용하여 상태 정보를 분산 배치함으로써, NAND타입의 플래시메모리와 같이 부분기록 회수에 제한이 있는 플래시메모리에 대해 블록이나 유닛의 기록상태를 관리할 수 있는 효과가 있다.

예를 들어 미정(1111)→유효(s0(1110)→s1(1100)→s2(1000)→s0(1110)→...)→삭제(0000)와 같은 절차로 변경할 수 있다. 따라서 NAND 타입 플래시메모리에 대해 이를 적용한다면 3회의 부분기록으로 데이터를 기록할 수 있다. 삭제된 상태인 경우에, 데이터와 논리적 블록번호 등 상태정보를 제외한 다른 데이터는 모두 무효한 것으로 가정한다면, 실제로 2회의 부분기록으로 기록연산을 수행하는 것과 같다.

삭제 유닛의 경우에도 첫 번째 블록에 논리적인 유닛 번호와 유효한 상태 여부, 이전 블록의 마모도(삭제 회수) 등을 기록하고, 두 번째 블록에 복사중/이동중 여부와 마모도 등을 기록한다면 3회의 부분 기록이면 충분하다. 상태 정보를 더 많은 블록에 분산시키거나 배치를 조절하면 부분 기록 회수를 더 줄일 수도 있다.

### (57) 청구의 범위

#### 청구항 1

적어도 1개의 블록을 구비하고 있는 유닛 단위로 영역이 구분된 플래시메모리의 재사상 제어 방법에 있어서,

(a)소정 블록에 대한 사상 정보를 토대로 소정의 물리적인 유닛을 찾는 단계;

(b)상기 (a)단계에서 찾아진 물리적인 유닛에 상기 소정 블록이 유효한 상태로 존재하면, 블록의 상태가 미정인 다른 블록을 찾는 단계;

(c)상기 다른 블록의 상태를 상기 소정 블록에 설정되어 있는 상태의 다음 상태로 변경하는 단계;

(d)상기 다른 블록에 새로운 데이터와 논리적 블록 번호와 같은 부가 정보를 기록하는 단계;

(e)상기 소정 블록의 상태를 삭제 상태로 변경하는 단계를 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

#### 청구항 2

제 1 항에 있어서, 상기 (c)단계는 4가지 상태중 하나의 상태로 설정된 상기 소정 블록에 따라 상기 다른 블록의 상태를 설정하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 3

제 2 항에 있어서, 상기 (c)단계는,

(c1)상기 소정 블록의 상태가 미정 상태이면, 상기 다음 블록의 상태를 유효 상태중 첫 번째 상태로 설정하는 단계;

(c2)상기 소정 블록의 상태가 유효 상태중 첫 번째 상태이면, 상기 다음 블록의 상태를 유효 상태중 두 번째 상태로 설정하는 단계;

(c3)상기 소정 블록의 상태가 유효 상태중 두 번째 상태이면, 상기 다음 블록의 상태를 유효 상태중 세 번째 상태로 설정하는 단계;

(c4)상기 소정 블록의 상태가 유효 상태중 세 번째 상태이면, 상기 다음 블록의 상태를 유효 상태중 첫 번째 상태로 설정하는 단계로 이루어지는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 4

제 1 항에 있어서, 상기 플래시메모리에 대한 재사상 제어방법은,

(f)상기 (a)단계에서 찾아진 물리적인 유니트에 상기 소정 블록이 존재하지 않거나 상기 소정 블록이 삭제 상태로 존재하면, 상기 소정 블록의 상태를 미정상태로 설정한 뒤, 블록의 상태가 미정인 다른 블록을 찾는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 5

제 5 항에 있어서, 상기 플래시메모리에 대한 재사상 제어방법은,

(g)상기 (f)단계 수행결과, 다른 블록이 존재하지 않으면, 상기 물리적인 유니트에 대해 재생연산을 수행하는 단계;

(h)상기 (g)단계 수행 후, 블록의 상태가 미정인 다른 블록을 다시 찾는 단계;

(i)상기 (h)단계 수행결과, 다른 블록이 찾아지면, 상기 (c)단계로 진행하는 단계;

(j)상기 (h)단계 수행결과, 다른 블록이 찾아지지 않으면, 오류로 처리하는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 6

제 5 항에 있어서, 상기 (g)단계는,

(g1)재생할 소스 유니트와 이동 유니트를 결정하는 단계;

(g2)상기 소스 유니트의 상태를 이동중으로 설정하고, 상기 이동 유니트의 상태를 복사중으로 설정하는 단계;

(g3)상기 소스 유니트내의 유효한 블록들, 관련 메타 및 헤더 정보를 복사하는 단계;

(g4)상기 복사중으로 설정되어 있는 이동 유니트를 유효 상태로 설정하면서 삭제 회수(마모도 값)를 1회 증가시키는 단계;

(g5)상기 소스 유니트를 삭제하면서 상기 (g4)에서 결정된 삭제 회수를 상기 소스 유니트에 기록하는 단계를 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 7

제 6 항에 있어서, 상기 (g)단계는,

(g6)상기 (g1)단계에서 결정된 상기 이동 유니트의 삭제 회수가 임계치를 넘는지를 체크하는 단계;

(g7)상기 (g6)단계의 체크결과, 상기 임계치를 넘지 않으면, 상기 (g2)단계로 진행하는 단계;

(g8)상기 (g6)단계의 체크결과, 상기 임계치를 넘으면, 상기 소스 유니트의 번호를 저장하는 단계;

(g9)상기 삭제 회수가 최소인 소스 유니트를 선택한 뒤, 상기 (g2)단계로 진행하는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 8

제 6 항에 있어서, 상기 (g)단계는,

(g6)상기 (g5)단계 수행 후, 상기 플래시메모리에 대한 사상 정보를 포함한 내부 자료 구조를 갱신하는 단계;

(g7)재생할 소스 유니트가 더 존재하는 지를 체크하는 단계;

(g8)상기 (g7)의 체크결과, 상기 재생할 소스 유니트가 더 존재하면, 상기 (g1)단계로 리턴하는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

### 청구항 9

제 6 항에 있어서, 상기 (g)단계는,

(g6)장애가 발생되면, 논리적인 유니트가 동일한 유니트가 두 개 존재하는 지를 체크하는 단계;

(g7)상기 (g6)단계의 체크결과, 하나는 이동중이고, 하나는 복사중인 동일한 유니트가 두 개 존재하면, 복사중인 유니트를 삭제하고 복사중이던 유니트에 삭제 회수를 기록하는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

#### 청구항 10

제 9 항에 있어서, 상기 (g)단계는,

(g8)장애가 발생된 상태에서 이동중인 상태로 설정된 유니트에 (g4)단계에서 생성된 삭제 회수가 기록되지 않은 상태이면, 상기 복사중인 유니트에 설정되어 있는 이동중인 유니트에 대한 삭제회수를 이동중인 유니트로 복사한 후, 재생 연산을 종료하는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

#### 청구항 11

제 1 항에 있어서, 상기 플래시메모리에 대한 재사상 제어방법은,

(f)장애가 발생되면, 하나의 논리적 블록 번호에 대해 유효한 상태인 블록이 두 개 존재하는 지를 체크하는 단계;

(g)상기 (f)단계 수행결과, 상기 유효한 상태의 블록이 하나만 존재하는 경우에는 블록 상태를 기록하는 과정을 종료하는 단계;

(h)상기 (f)단계 수행결과, 상기 유효한 상태의 블록이 두 개 존재하는 경우에, 두 블록의 블록상태를 체크하여 상기 소정 블록(이전 블록)과 다음 블록(새 블록)을 구분하는 단계;

(i)상기 소정 블록과 다음 블록중 하나의 블록을 삭제하는 단계를 더 포함하는 것을 특징으로 하는 플래시메모리에 대한 재사상 제어방법.

#### 청구항 12

적어도 1개의 블록을 구비하고 있는 유니트 단위로 영역이 구분된 플래시메모리에 있어서,

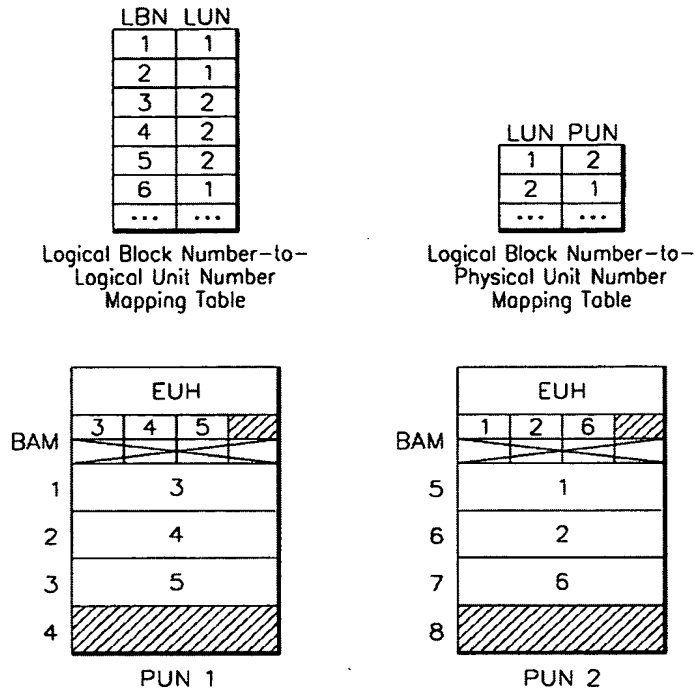
상기 유니트의 첫 번째 블록은 논리적인 유니트 번호와 유효한 상태여부, 이전 블록의 삭제 회수를 포함하는 정보가 기록되도록 할당하고,

상기 유니트의 두 번째 블록은 복사중/이동중 여부와 삭제 회수(마모도)를 포함하는 정보가 기록되도록 할당하도록 구성되고,

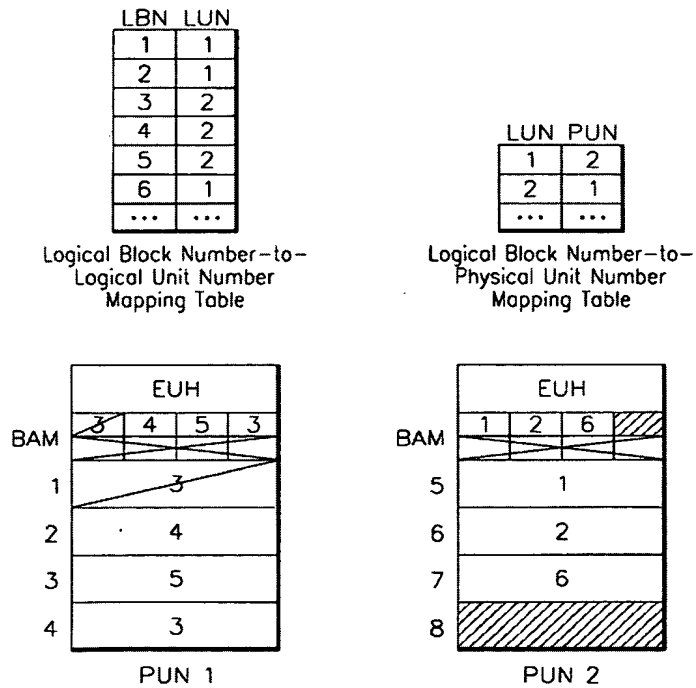
상기 유니트의 데이터가 기록되는 블록의 여유 어레이의 소정 영역에는 순환계수 방식으로 결정된 블록 상태를 나타내는 정보가 기록되도록 구성된 플래시메모리.

도면

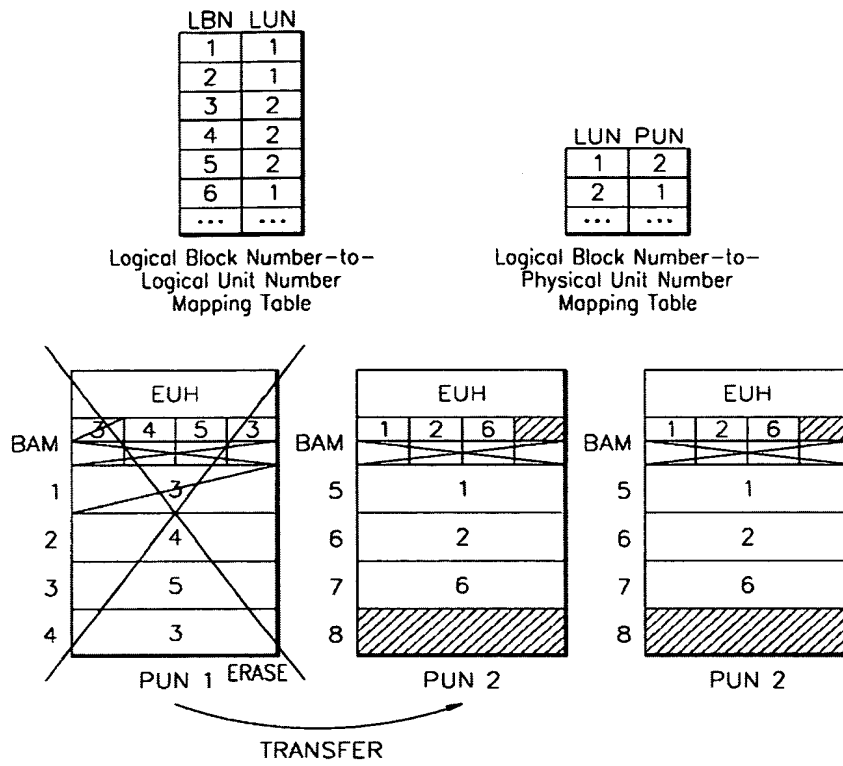
도면1



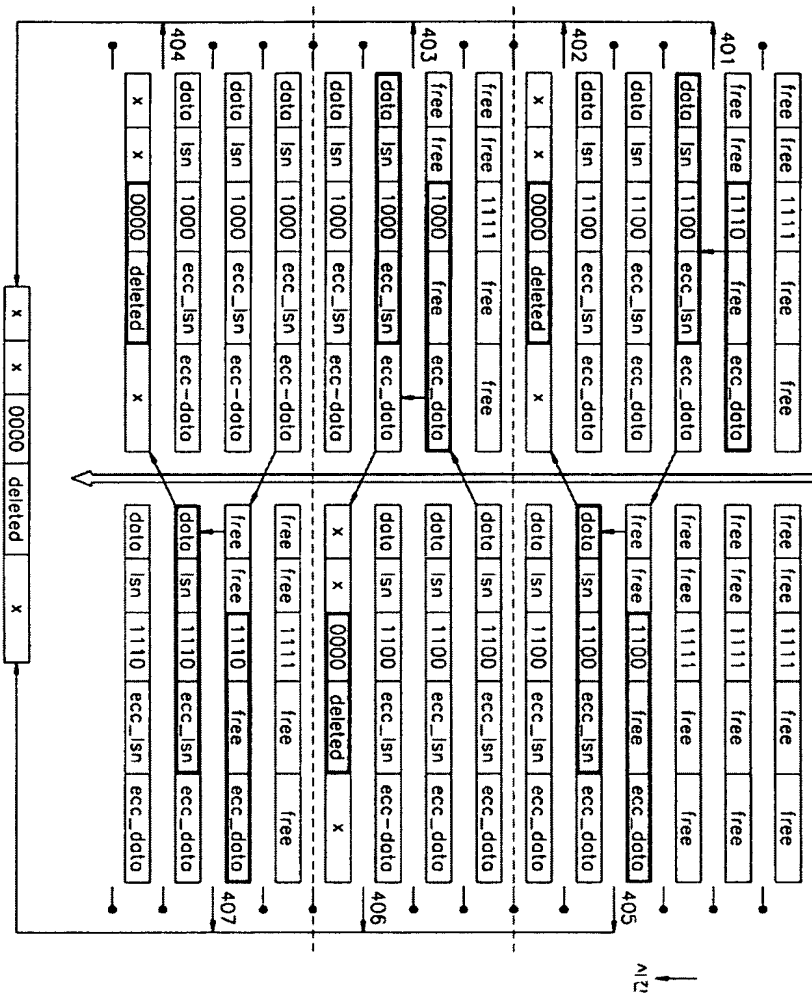
도면2



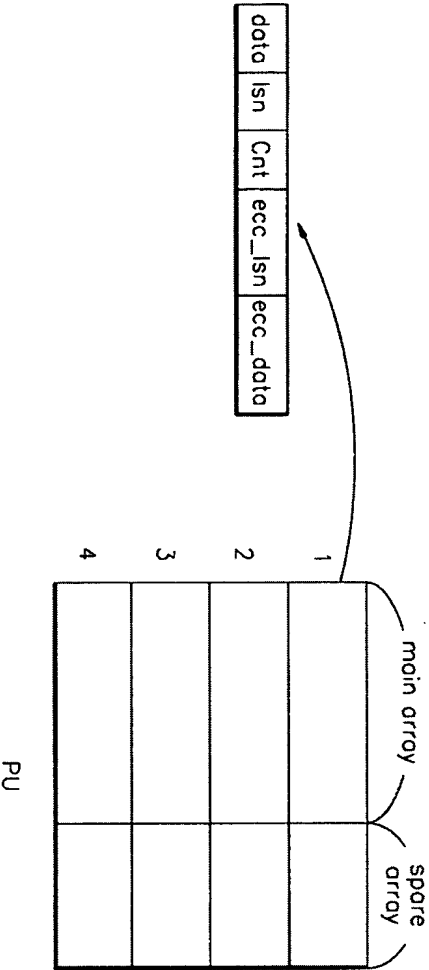
도면3



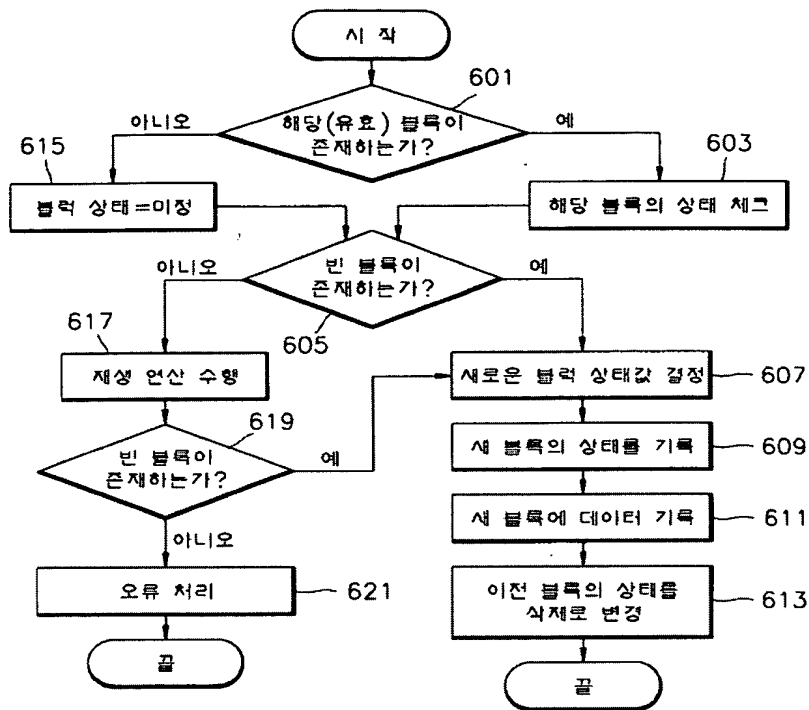
도면4



도면5



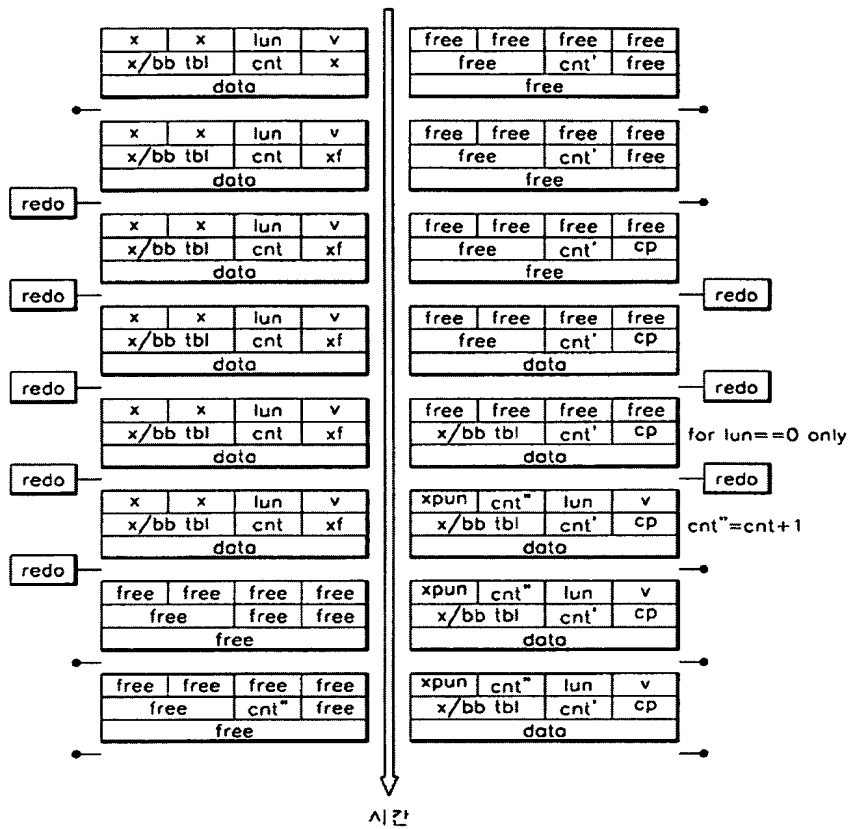
도면6



도면7

xpun	xcnt	lun	v
x/bb tbl	cnt	cp/xt	
data			

도면8



도면9

